Agent Based Software Engineering

LabTask 04

Name : Muhammad Shamoil
Section : BSSE – V – A
Date: 23rd Oct , 2023
Submitted To : Sir Waseem Wahid

Task 01

```python
import random

# Winning State
goal_state = [[1, 2, 3],
              [4, 5, 6],
              [7, 8, 0]]

def generate_puzzle(): #Generation of random puzzle
    nums = list(range(9)) # Numbers 0 to 8
    random.shuffle(nums)
    return [nums[0:3], nums[3:6], nums[6:9]]

def find_zero(state): #locating zero state or empty space
    for i in range(3):
        for j in range(3):
            if state[i][j] == 0:
                return i, j
    return

def print_puzzle(state): #Print puzzle
    print()
    for row in state:
        print("  ".join(str(num) if num != 0 else " " for num in row))
    print()

def move(state, direction): #Control of empty state
    i, j = find_zero(state)
    new_state = [row[:] for row in state]

    if direction == "up" and i > 0:
        # Swap with tile above
        new_state[i][j], new_state[i-1][j] = new_state[i-1][j], new_state[i][j]

    elif direction == "down" and i < 2: # Boundary changed from 1 to 2
        # Swap with tile below
        new_state[i][j], new_state[i+1][j] = new_state[i+1][j], new_state[i][j]

    elif direction == "left" and j > 0:
        # Swap with tile to the left
        new_state[i][j], new_state[i][j-1] = new_state[i][j-1], new_state[i][j]

    elif direction == "right" and j < 2: # Boundary changed from 1 to 2
        # Swap with tile to the right
        new_state[i][j], new_state[i][j+1] = new_state[i][j+1], new_state[i][j]

    else:
        print("Invalid move! Try again.")

    return new_state

print("--- 3x3 Sliding Puzzle ---")
puzzle = generate_puzzle()

while puzzle != goal_state:
    print_puzzle(puzzle)
    move_direction = input("Move (up/down/left/right): ").strip().lower()

    if move_direction in ["up", "down", "left", "right"]:
        puzzle = move(puzzle, move_direction)
    else:
        print("Invalid input. Please enter 'up', 'down', 'left', or 'right'.")

print("\n--- Solved! ---")
print_puzzle(puzzle)
print("Congratulations! You solved it!")
```

```
--- 3x3 Sliding Puzzle ---

5  1  6
2     7
4  3  8

Move (up/down/left/right): up

5     6
2  1  7
4  3  8

Move (up/down/left/right): down

5  1  6
2     7
4  3  8

Move (up/down/left/right): right

5  1  6
2  7
4  3  8

Move (up/down/left/right): left

5  1  6
2     7
4  3  8

Move (up/down/left/right): bottom
Invalid input. Please enter 'up', 'down', 'left', or 'right'.
```

Task 02

```python
#cities and their direct connections
cities = {
    "Murree": ["Islamabad"],
    "Islamabad": ["Murree", "Rawalpindi"],
    "Rawalpindi": ["Islamabad", "Kallar Kahar"],
    "Kallar Kahar": ["Rawalpindi", "Chakwal"],
    "Chakwal": ["Kallar Kahar", "Sargodha"],
    "Sargodha": ["Chakwal", "Faisalabad"],
    "Faisalabad": ["Sargodha", "Lahore"],
    "Lahore": ["Faisalabad", "Multan"],
    "Multan": ["Lahore", "Sukkur"],
    "Sukkur": ["Multan", "Hyderabad"],
    "Hyderabad": ["Sukkur", "Karachi"],
    "Karachi": [] # Destination
}

# Start and goal cities
current_city = "Murree"
goal_city = "Karachi"

print("Welcome to the City Path Finder Game!")
print(f"You are starting in {current_city}. Try to reach {goal_city}.\n")

# --- Game loop ---
while current_city != goal_city:
    print(f"You are currently in: {current_city}")

    # Check for dead end
    if not cities[current_city]:
        print("You are at a dead end, but it's not the goal!")
        break

    print(f"Possible cities to move to: {', '.join(cities[current_city])}")

    # input
    move = input("Where would you like to move? ").strip()

    # move checking
    if move in cities[current_city]:
        current_city = move
        print(f"Moving to {current_city}...\n")
    else:
        print("Invalid move! You can only move to cities directly connected to your current city.\n")

# --- End of game ---
if current_city == goal_city:
    print(f"Congratulations! You reached {goal_city}!")
```

```
Welcome to the City Path Finder Game!
You are starting in Murree. Try to reach Karachi.

You are currently in: Murree
Possible cities to move to: Islamabad
Where would you like to move? Islamabad
Moving to Islamabad...

You are currently in: Islamabad
Possible cities to move to: Murree, Rawalpindi
Where would you like to move? Rawalpindi
Moving to Rawalpindi...

You are currently in: Rawalpindi
Possible cities to move to: Islamabad, Kallar Kahar
Where would you like to move? Kallar Kahar
Moving to Kallar Kahar...

You are currently in: Kallar Kahar
Possible cities to move to: Rawalpindi, Chakwal
Where would you like to move? chakwal
Invalid move! You can only move to cities directly connected to your current city.

You are currently in: Kallar Kahar
Possible cities to move to: Rawalpindi, Chakwal
Where would you like to move? Chakwal
Moving to Chakwal...

You are currently in: Chakwal
Possible cities to move to: Kallar Kahar, Sargodha
Where would you like to move? sarghoda
Invalid move! You can only move to cities directly connected to your current city.

You are currently in: Chakwal
Possible cities to move to: Kallar Kahar, Sargodha
Where would you like to move? Sargodha
Moving to Sargodha...

You are currently in: Sargodha
Possible cities to move to: Chakwal, Faisalabad
Where would you like to move? Faisalabad
Moving to Faisalabad...

You are currently in: Faisalabad
Possible cities to move to: Sargodha, Lahore
Where would you like to move? Lahore
Moving to Lahore...

You are currently in: Lahore
Possible cities to move to: Faisalabad, Multan
Where would you like to move? Multan
Moving to Multan...

You are currently in: Multan
Possible cities to move to: Lahore, Sukkur
Where would you like to move? Sukkur
Moving to Sukkur...

You are currently in: Sukkur
Possible cities to move to: Multan, Hyderabad
Where would you like to move? Hyderabad
Moving to Hyderabad...

You are currently in: Hyderabad
Possible cities to move to: Sukkur, Karachi
Where would you like to move? Karachi
Moving to Karachi...

Congratulations! You reached Karachi!
```

Task 03

```python
def simulate_and_gate(): #AND SIMULATION
    print("\n--- AND Gate: Office Door Access ---")
    print("You must have BOTH a keycard AND the correct PIN.")

    has_keycard = input("Do you have the keycard? (yes/no): ").strip().lower()
    knows_pin = input("Do you know the PIN? (yes/no): ").strip().lower()

    # AND condition: Both must be true
    if has_keycard == "yes" and knows_pin == "yes":
        print(">>> Access Granted. Welcome!")
    else:
        print(">>> Access Denied. Both a keycard and PIN are required.")

def simulate_or_gate(): #OR SIMULATION
    print("\n--- OR Gate: Street Lights Control ---")
    print("Lights turn on if it's dark OR if the manual override is on.")

    is_dark = input("Is it dark outside? (yes/no): ").strip().lower()
    is_manual_override = input("Is the manual override switch on? (yes/no): ").strip().lower()

    # OR condition: Either one (or both) can be true
    if is_dark == "yes" or is_manual_override == "yes":
        print(">>> Street lights are ON.")
    else:
        print(">>> Street lights are OFF.")

#-----RUN-----------
simulate_and_gate()
simulate_or_gate()
```

```
--- AND Gate: Office Door Access ---
You must have BOTH a keycard AND the correct PIN.
Do you have the keycard? (yes/no): yes
Do you know the PIN? (yes/no): no
>>> Access Denied. Both a keycard and PIN are required.

--- OR Gate: Street Lights Control ---
Lights turn on if it's dark OR if the manual override is on.
Is it dark outside? (yes/no): no
Is the manual override switch on? (yes/no): yes
>>> Street lights are ON.
```